Exhibit B

AGENT FACTORY TECHNOLOGY

The Agent factory comprises a plurality of computer algorithms that perform the various functions described below. Each artificial agent is a prediction algorithm designed to recognize and exploit predictability bubbles. The agent factory software routines together train the agents and determine which agents should be provided to users.

The agent factory comprises 12 functional software routines:

1) data,
2) run agents,
3) refresh,
4) trades,
5) portfolio,
6) test,
7) quality,
8) distribute files to user database,
9) temp,
10) select,
11) A-brain,
12) B-brain,

The agent factory preferably maintains two pools of agents: active agents that are made available to users and reserve agents that preferably are used to replace active ones when they are retired as a result of changing market conditions. The main functions fall in two groups: management of active or live agents, and production of new reserve agents.

First group: managing the pool of live agents

"Data" reads new daily market price data from a financial data provider and makes this information available to the other factory functions.

"Run agents" updates each existing agent's recommendations in light of the new data and retires agents that have a negative predictability value.

"Refresh" looks for agents that have been marked as "retired" due to changing market conditions, places these in a "cemetery" database and replaces them with reserve agents (previously generated) that have a high predictability value and trading strategies that are diverse to those currently adopted by existing active agents, usually because they are using different prediction horizons or templates.

"Trades" looks for agent consensus decisions using various non-linear consensus rules that identify teams of two or more agents that agree to recommend a trade. For example, if there exists for a given stock an agent with a short term horizon and positive PFI (a so-called "market timing" agent) together with a medium to long term horizon agent and both with predictability above a certain threshold and there exists no similar agent with a contrary opinion, then a position will be opened if both agents are in agreement. Trades also reviews trading positions recommended by teams of agents in previous runs of the factory process and decides if such positions should be closed based on the new data.

"Portfolio" looks for agent consensus decisions based on a formula to arrive at a consensus decision taking into account the opinions of all agents. For example, this consensus can be derived from a risk and reward analysis. In this implementation, a consensus force is the weighted average of the positions recommended by each agent in that stock (e.g., 1 for a buy recommendation, 0 for neutral and –1 for a sell recommendation), where the weights are determined by an iterative algorithm that seeks to maximize the ratio of the portfolio's yield to its volatility. "Portfolio" manages a set of virtual portfolios that calculate their positions in each stock from the consensus force signal, for example using a proportionality rule, and apply different investor strategies, including, preferably, both investor portfolios and leveraged derivative portfolios.

"Test" measures performance statistics by combining data from the cemetery database, an active agents database, a Portfolio database and a database of consensus trade decisions, such as the average profit earned by each agent in the past month and the average profit per share traded using different consensus trading rules, or other similar comparison paradigms.

"Quality" checks that each parameter of every active and reserve agent has the appropriate format and is within lower and upper bounds that ensure that they will perform according to a user's expectations.

Second group: creating new reserve agents

"Temp" is a template analyzer that tests the effectiveness of different technical trading rules on a window of historical data and evaluates the predictability of each trading rule.

"Select" looks for predictable trading rules from those discovered by temp and selects a subset of these that are most predictable and satisfy diversity enforcement criteria that, for example, avoid repeating rules that are already being used by previously-created agents.

"A-brain" creates a preliminary artificial agent that uses the technical trading rule selected and generates a preprocessed signal (chart) available for further processing.

"B-brain" iterates a process that implements a pre-trained neural network from a library of candidates and completes the agent's training by letting it learn how to interpret the preprocessed signal generated by "A-brain"; for each choice of neural network it creates an agent and analyzes its predictability. The most predictable agent from these trials is kept in a database to be later "packaged" and sent to the ultimate user. Among the B-brains in this library are some candidates that apply strategies that would be recognized as common technical trading rules and at times other candidates that apply more complex interpretations of the pre-processed signals from the A-brain, including contextual learning and backpropagation neural networks.

The explicit codes included are:

```
2roa.c
a0330.c
abracadabra.c
abm0330.c
act_dividendos.c
act_emisoras.c
action.c
albl0330.c
atecmi.c
b0330.c
build.c
bury0330.c
category.c
ctst0330.c
data_comp.c
data0330.c
datos.c
fabrica.c
fastrun
fechas.c
firstrun
frsh0330.c
ftd0330.c
genesis
initialize.c
landscape.c
la0330.c
lb0330.c
mkdata.c
neur0330.c
parameters.c
pflo0330.c
prep0330.c
prst0330.c
qlty0330.c
slct0330.c
stat0330.c
temp0330.c
tgt0330.c
tneu0330.c
```

```
tomo0330.c
trds0330.c
rtst0330.c
wintolinux.c
backprop.c
burry_agt.c
choose_emisoras.c
contact.c
datavectors.c
definiciones.c
dump_register.c
dump_trds_register.c
evaluate_model.c
find_daynumber.c
find_validez.c
input_agents.c
input_data.c
input_osc.c
input_register.c
input_val.c
load_names.c
load_status_a.c
load_status_b.c
load_status_la.c
load_status_lb.c
load_targets.c
past_decisions.c
plines.c
recommend.c
reset_arrays.c
screen.c
see_results.c
set_markers.c
set_screen.c
stat_screen.c
status_bits.c
update_register.c
vlt.c
vlt_B.c
add_relevant.c
choose_markers.c
dump_screen.c
dynamic_markers.c
get_current_x.c
histogram.c
load_LR.c
```

load_relevant.c
load_rs.c
load_screen.c
load_section.c
refine_screen.c
reset_arrays.c
see_results.c
variable_names.c

All these codes are written in the C programming language.

GUI codes

Adaptive_lan.vbp
Adaptive_lan__nasdaq.vbp

These are Visual Basic projects written in Visual Basic 6.0

Windows based agent preparation codes

mi.vbp
borrador.vbp
Prepara_entrega.c

The date of last revision for all of the above codes is May 1, 2001.